Claims

1. Method for the compressed transmission and/or storage of a text represented by digital data, the structure of which is defined by a grammar, characterized in that

   a) prior to storage and/or transmission the text is converted into a linear sequence of symbols that specify the successive application of grammatical rules to form the text, and

   (b) to decode the text after the storage and/or transmission, for every symbol thus produced by syntax-directed coding (SDC) the grammatical rule corresponding to that particular SDC symbol is performed and particular output data are generated by this rule.

2. Method according to Claim 1, characterized in that to generate the SDC symbols

   (a) by parsing the text a parse tree corresponding to the consecutive applications of the grammatical rules is generated,

   (b) to each node in this parse tree there is attributed an SDC symbol, which out of all the rules permitted by the grammar at this site unambiguously identifies the particular rule that is actually used to generate the original text, and

(c) the SDC symbols are concatenated, according to a fixed order of traversing all nodes of the parse tree, to form a linear sequence of SDC symbols.

3. Method according to Claim 1, characterized in that for decoding

(a) the uppermost entry in a stack memory is replaced according to the grammar production determined by the SDC symbol that has been input,

(b) parts of the grammatical rules that cannot yet be completely processed are deposited in a stack memory, and

(c) parts of the production for which substitution is complete are output as part of the original text.

4. Method according to Claim 1, characterized in that for decoding

(a) the uppermost entry in a stack memory is replaced according to the grammar production determined by the SDC symbol that has been input,

(b) parts of the grammatical rules that cannot yet be completely processed are deposited in a stack memory, and

(c) parts of the production for which substitution is complete are processed immediately to form part of an executable program for a real processor or a virtual machine.

5. Method according to Claim 3 or 4, characterized in that a stack machine

(a) is initialized by depositing a specific start symbol (non-terminal symbol) into the empty stack memory,

(b) reads the uppermost symbol from the stack memory,

(c) checks whether the read symbol is a terminal or a non-terminal symbol,

(d) if it is a terminal symbol, outputs the symbol and, depending on whether additional symbols are present in the stack memory, either continues with 5b or terminates if the stack memory is empty, or

(e) if it is a non-terminal symbol, reads the next SDC symbol from the input stream,

(f) depending on what SDC symbol has been read, selects precisely one alternative (chain of terminal and/or non-terminal symbols) out of the set of alternatively applicable replacement rules (productions) that are valid

for the non-terminal symbol currently being processed, and

(g) places this chain of terminal and/or non-terminal symbols into the stack memory and continues with 5b.

6. Method according to Claim 2, characterized in that

(a) to each node in the parse tree there is attributed an SDC symbol and the probability distribution of all the SDC symbols possible in this node,

(b) the linear sequence of SDC symbols is subjected to entropy encoding in conformity with the associated probability distributions,

(c) the entropy decoding is carried out with a probability distribution of SDC symbols identical to that used for the entropy encoding.

7. Method according to Claim 6, characterized in that

(a) the probability distribution of the rules that can be applied in a node, starting from an initial distribution, is adapted at each appearance of an SDC symbol in such a way that the probability of the SDC symbol that appears is increased and the probability of all other

symbols is correspondingly reduced,

(b) the currently valid distribution of occurrence probabilities is assigned to the SDC symbols of the associated node type,

(c) the probability distribution of all SDC symbols in a current node, together with the SDC symbol to be encoded, forms a model for an arithmetic encoding,

(d) during decoding the end of the text is recognized by the fact that the stack memory is empty, and

(e) an End-Of-Message (EOM) symbol required for arithmetic coding is eliminated.

8. Apparatus for the method according to Claim 1, characterized by an encoder comprising

(a) a scanner to transform text comprising a sequence of readable characters into a sequence of terminal symbols,

(b) a parser to find grammatical rules, the successive application of which was originally used to generate the sequence of terminal symbols,

(c) a mapper, which unambiguously associates syntax-directed symbols with the rules identified by the parser and outputs these

symbols in a fixed sequence,

and a decoder comprising

(a) a stack machine which, according to the uppermost symbol in the stack memory and, where appropriate, the adjacent SDC symbol, outputs the already fixed terminal symbol, or deposits the sequence of terminal and/or non-terminal symbols associated with the current symbol into the stack memory, and

(b) a lexicon that replaces the terminal symbols by chains of readable alphanumeric characters.

9. Apparatus for the method according to Claim 3, characterized by an encoder comprising

a) a scanner to transform a program that is present in a source text or in a form derived from the source text by a preprocessor, into a sequence of terminal symbols,

(b) a parser to find the grammatical rules, the successive application of which was originally used to generate the sequence of terminal symbols, and

(c) a mapper, which unambiguously associates syntax-directed symbols with the rules identified by the parser and outputs these symbols in a fixed sequence,

and a decoder comprising

(a) a stack machine which, according to the uppermost symbol in the stack memory and, where appropriate, the adjacent SDC symbol, outputs the already fixed terminal symbol, or deposits the sequence of terminal and/or non-terminal symbols associated with the current symbol into the stack memory, and

(b) a code generator that generates from the sequence of terminal symbols executable machine code, or intermediate code to be executed on a virtual machine.

10. Apparatus for the method according to Claim 7, comprising

(a) on a transmitter side

i. a table that contains the probability distributions of the SDC symbols for each node type, the contents of which are established at initialization with fixed initial probability distributions for each node type,

ii. an adapter that updates the probability distribution of the SDC symbols for the node type valid at any given moment with reference to the existing probability distribution, the SDC symbol to be encoded and the current node type, and enters this new probability distribution

into the table, and

iii. an arithmetic coder that encodes each SDC symbol to be encoded with the currently valid probability distribution supplied by the adapter;

(b) and on a receiver side

i. a table that contains the probability distributions of the SDC symbols for each node type, the contents of which are established at initialization with fixed initial probability distributions for each node type,

ii. an adapter that updates the probability distribution of the SDC symbols for the node type valid at any given moment, as established by the stack machine, with reference to the existing probability distribution, the SDC symbol to be encoded and the current node type, and enters this new probability distribution into the table,

iii. an arithmetic decoder which, with reference to the currently valid probability distribution of the current node type supplied by the adapter, decodes the next SDC symbol and sends it to the stack machine for further processing.